Available online at www.sciencedirect.com
ScienceDirect
PERGAMON
Acta Astronautica 65 (2009) 1616–1627
ACTA ASTRONAUTICA
www.elsevier.com/locate/actaastro

FPGA-based operational concept and payload data processing for the Flying Laptop satellite

Toshinori Kuwahara[a,*], Felix Böhringer[a], Albert Falke[a], Jens Eickhoff[b], Felix Huber[c], Hans-Peter Röser[a]

[a] *Institute of Space Systems, Universität Stuttgart, Pfaffenwaldring 31, 70569 Stuttgart, Germany*
[b] *EADS Astrium GmbH, 88039 Friedrichshafen, Germany*
[c] *Steinbeis Transferzentrum Raumfahrt, Rötestr. 15, 71126 Gäufelden, Germany*

Received 24 January 2009; received in revised form 5 April 2009; accepted 14 April 2009
Available online 22 May 2009

Abstract

Flying Laptop is the first small satellite developed by the Institute of Space Systems at the Universität Stuttgart. It is a test bed for an on-board computer with a reconfigurable, redundant and self-controlling high computational ability based on the field programmable gate arrays (FPGAs). This Technical Note presents the operational concept and the on-board payload data processing of the satellite. The designed operational concept of Flying Laptop enables the achievement of mission goals such as technical demonstration, scientific Earth observation, and the payload data processing methods. All these capabilities expand its scientific usage and enable new possibilities for real-time applications. Its hierarchical architecture of the operational modes of subsystems and modules are developed in a state-machine diagram and tested by means of MathWorks Simulink-/Stateflow Toolbox. Furthermore, the concept of the on-board payload data processing and its implementation and possible applications are described.

Keywords: Data handling; Image processing; FPGA; Operational design; Small satellite

1. Introduction

The small satellite *Flying Laptop* is the first satellite within the "Stuttgart Small Satellite Program" in which the Institute of Space Systems (IRS: Institut für Raumfahrtsysteme) at the Universität Stuttgart is developing several small satellites aiming to launch a small spacecraft to the Moon (*Lunar Mission BW*1 [1]) as the final goal. All these satellites are developed, built and operated by the IRS. The mission objectives of *Flying Laptop* are new technology demonstrations—partly for the following satellites within this program—and scientific Earth observation with its three payload camera systems and high speed communication systems as scientific instruments. *Flying Laptop* is a cubic, 3-axis stabilized satellite with the edge lengths of about 600 mm×700 mm×800 mm and a mass of about 120 kg. Detailed system design and mission facts can be seen in [2].

2. Operational design

2.1. Operation concept

The *Flying Laptop* has a number of mission objectives which can be divided into the following

* Corresponding author. Tel.: +49 711 685 69611.
E-mail address: kuwahara@irs.uni-stuttgart.de (T. Kuwahara).

0094-5765/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.actaastro.2009.04.011

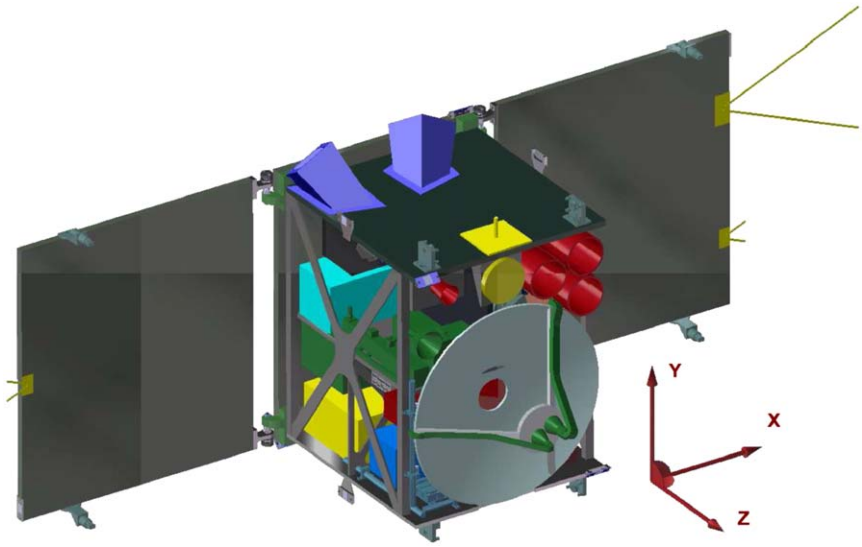

Fig. 1. *Flying Laptop* satellite mechanical configuration.

two categories:

I: New technology demonstrations

- Field programmable gate array (FPGA) based on-board computer (OBC) system.
- Ka-band bi-directional communication (up to 500 Mbps).
- High performance attitude control system (absolute pointing accuracy of better than 50 arcsec).
- Electrical solar panel deployment mechanism.
- GENIUS-GPS experiment (attitude determination by means of GPS signals).
- Near Earth Asteroid detection (star tracker).
- "Rent-A-Sat Mode" (lent the satellite to third parties to evaluate their own software on-board).
- Space-use demonstration of several components, e.g., fiber optical gyros, solar cells, etc.

II: Scientific Earth observation

- Multi-spectral Earth observation.
- Thermal infrared Earth observation.
- Bidirectional reflectance distribution function (BRDF) measurement [3].
- Precipitation measurement (Ka/Ku-band).
- Atmospheric attenuation measurement (Ka-band).
- Continuous Earth surface scanning.

In order to achieve these wide-ranging mission objectives, the operation of the satellite needs to be supported by discrete time-to-time planning rather than continuous and periodical operational planning, partly because most of the missions are single-shot missions. For example, a BRDF measurement of a specific target at a specific time shall be planned by analyzing the availability of the required sensors such as star trackers according to the satellite position, target position, and sun incident angle. Furthermore, supported by the capability of reconfigurable computing based on the FPGA technology, some of the scientific missions are still under detailed analysis and design for later implementations. Therefore, the operational plan shall be flexible and adaptable in late mission phases even after the launch. In order to achieve this operational concept, the commanding function shall support any kind of combination of satellite operations. In this way, the mission operation plan can be continuously updated in consequence of individual mission/experiment planning.

The other aspect of the difficulty in the mission planning of the *Flying Laptop* is that its on-board computer is purely based on the FPGA technology, which means there is no traditional processor aboard. The functionalities which are usually implemented into software on other satellites need to be replaced with a complex hardware logic realized in the FPGA chip. The operational plan shall be designed so that, on the one hand, the strong capability of parallel processing of the FPGA can be maximized and, on the other hand, the operability of the satellite system by telecommanding can ensure trouble-free conduction of every single mission.

Table 1
Flying Laptop system configuration.

Subsystem	Components
Payload (P/L)	Multi-spectral imaging camera system (MICS) Thermal imaging camera system (TICS) Panoramic camera system (PAMCAM) Ka-band communication system Ku-band transmitter
Attitude control system (ACS)	Reaction wheels Magnetic torquers Magnetometers GPS Star tracker Fiber optical gyros Sun sensors
Telemetry, tracking and command	S-band communication system UHF-band communication system VHF-band receiver
Data handling	On-board computer (OBC) Ultra stable oscillator
Thermal control system	House keeping unit Heaters Heat pipe Radiators
Power supply system	Power control and distribution unit Battery management system Solar panels (right, middle, left) Battery
Mechanisms	Solar panel deployment mechanism

2.2. Model design

Generally, satellite peripherals can be grouped into several subsystems hierarchically. That means the satellite system can be decomposed into several subsystems and those subsystems can be further decomposed into components. In this way, a hierarchical relationship between satellite system, subsystem, and components can be established. A graphical representation of the *Flying Laptop* satellite is illustrated in Fig. 1 and its subsystems and components are listed in Table 1.

In order to get the greatest benefit of the parallel processing feature of the FPGA, each subsystem is controlled completely in parallel to each other. This concept is enabled by the allocation of a state-machine to each of the subsystems and their belonging components. A state-machine is a diagram which illustrates the relationship between several modes and the transition conditions. Consequently, a system mode can be described as a combination of subsystem modes and a subsystem mode can be described as a certain combination of component modes which belong to the subsystem. From the technical point of view, in terms of hardware logic design within the FPGA, these application algorithms can be implemented as completely parallel processes in the subsystem level and component level. These parallel processes are controlled by individual commands. Finally, as the result of interaction between all subsystems (as the result of combinational commanding in other words), the system can perform desired functions.

The designed system level state-machine is illustrated in Fig. 2, and the state-machine of the attitude control system is illustrated in Fig. 3 as an example of one of the subsystem state-machines. Fig. 4 illustrates the pointing modes of the attitude control system. The designed pointing modes enable the desired observation capability of the satellite [4].

2.3. Design environment

For the design procedure, the following two aspects play important roles: mode design and transition logic design. The former ensures that there are enough modes provided with which the system, subsystems and, components can fulfill their functional requirements as described in Chapter Section 2.2. The latter defines possible transition paths between all modes and specifies triggering events and conditions for all of those transition paths. This transition logic shall be designed in such a way that the consistency between all subsystems can always be ensured. Generally, a system with i subsystems has $i(i-1)$ interfaces between them, and a subsystem with j modes has $j(j-1)$ theoretically possible transition paths between the modes. This large number of combinational states shall always be consistent.

In this study, the MathWorks Simulink-/Stateflow Toolbox is applied for designing the system, subsystem and component modes of the *Flying Laptop*. Fig. 5 illustrates the primary Simulink model which includes four subsystem Stateflow models. These Stateflow models communicate with each other and exchange current mode information. In order to let a state transition in a Stateflow model trigger consequential state transitions in other Stateflow models, the triggering Stateflow model generates event signals, and delivers them to the other models. Some of the triggered models transit to another mode and also generate event

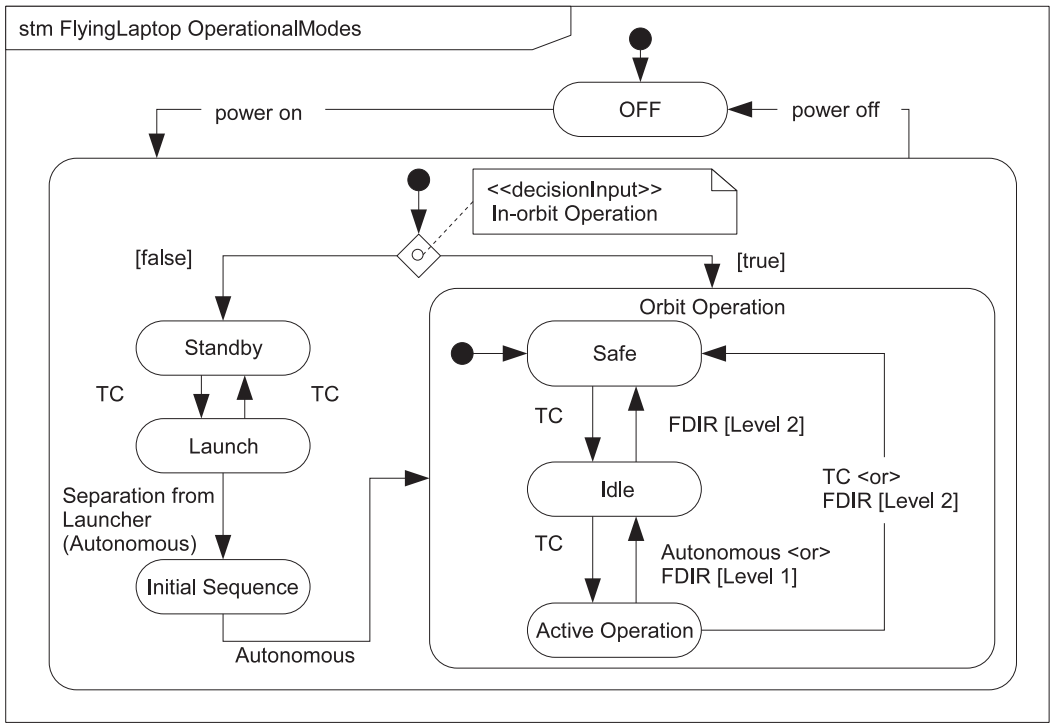

Fig. 2. System level state-machine.

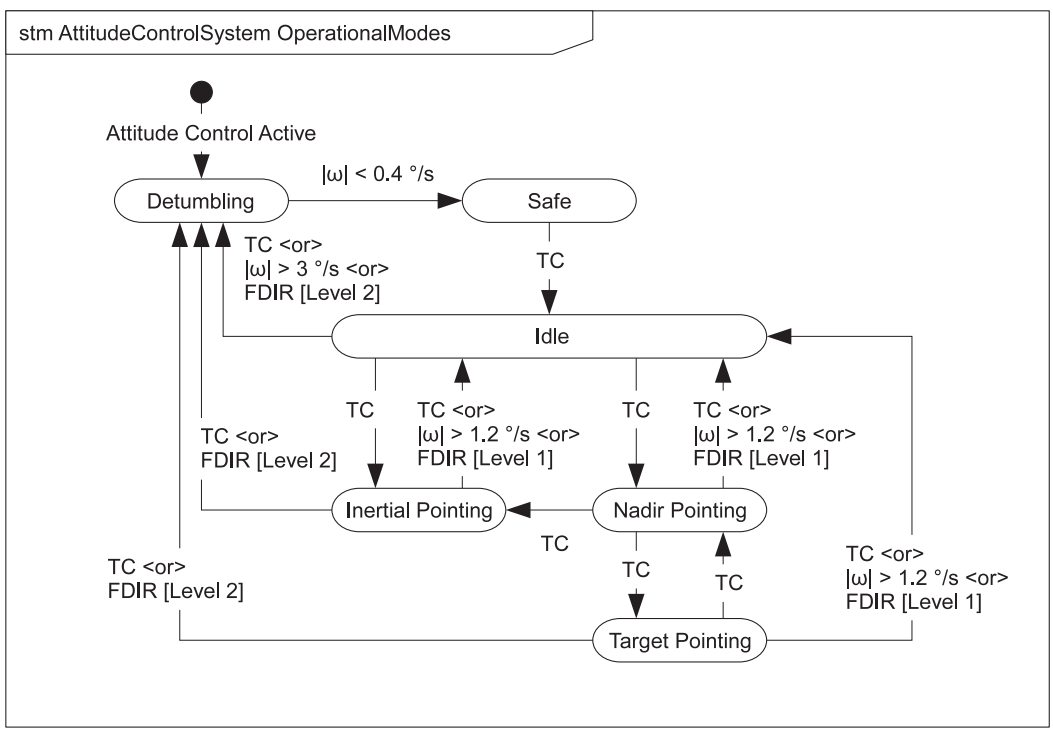

Fig. 3. Attitude control system state-machine.

signals. In the end, all models come to a consistent mode combination as the result of iterative interaction between each other.

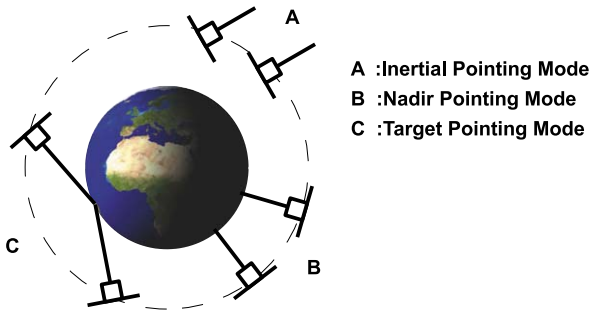

Fig. 4. Pointing modes of attitude control system.

The internal structure of the attitude control system Stateflow model is illustrated as an example in Fig. 6. As is illustrated in this figure, the Stateflow Toolbox allows to define the possible transition path as a single directional arrow between two states, and also one can define the transition conditions from one state to another allocating them to the specific transition path. In addition to this, each Stateflow model can contain internal variables and pre-defined logic-bit-tables so that allowed transition mask tables can be implemented. It also supports execution of variable manipulation at three different phases: entering phase into a state, maintaining phase in a state and moving-out phase from a state. These functions provide a convenient development environment.

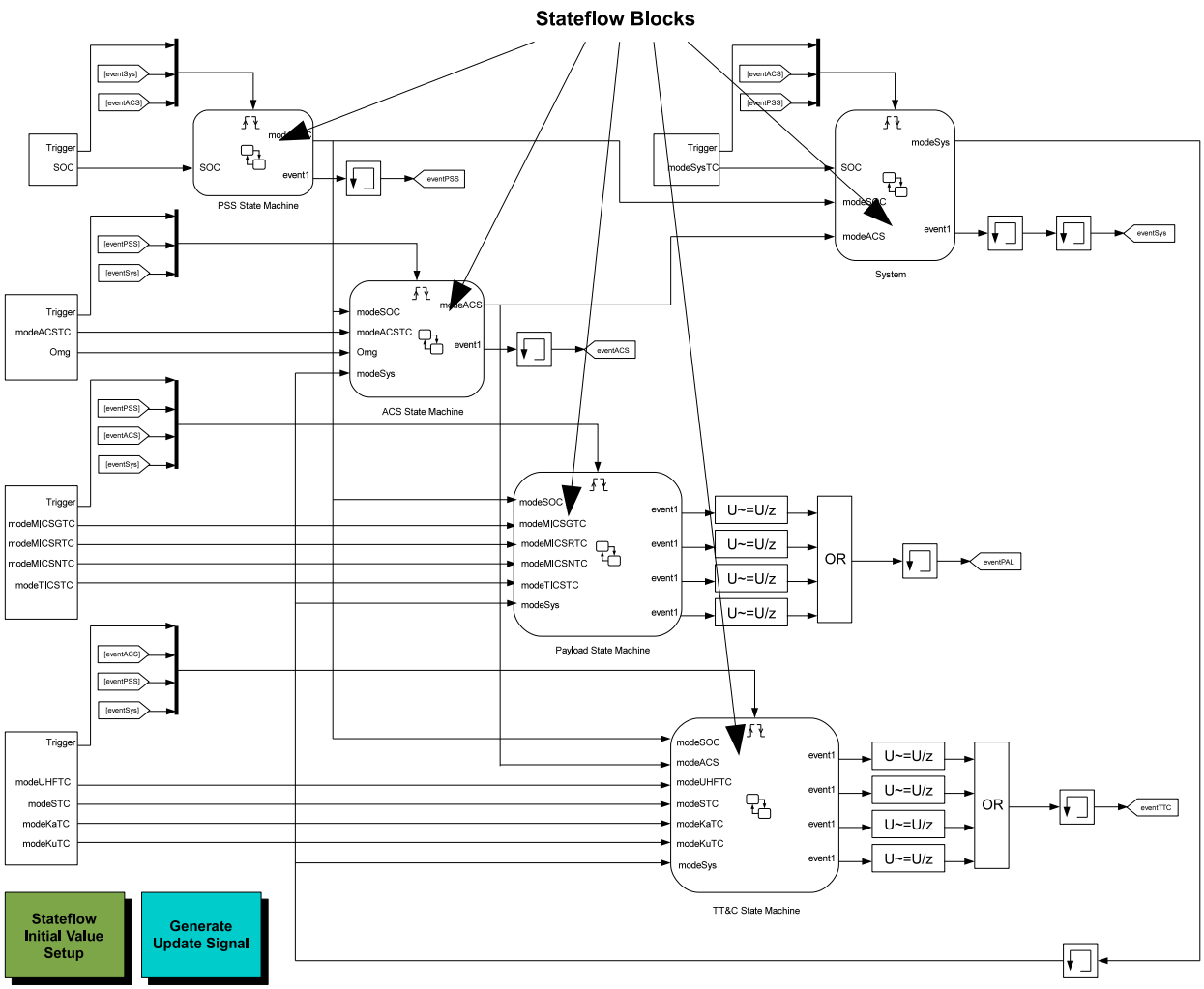

Fig. 5. State-machine model of *Flying Laptop*.

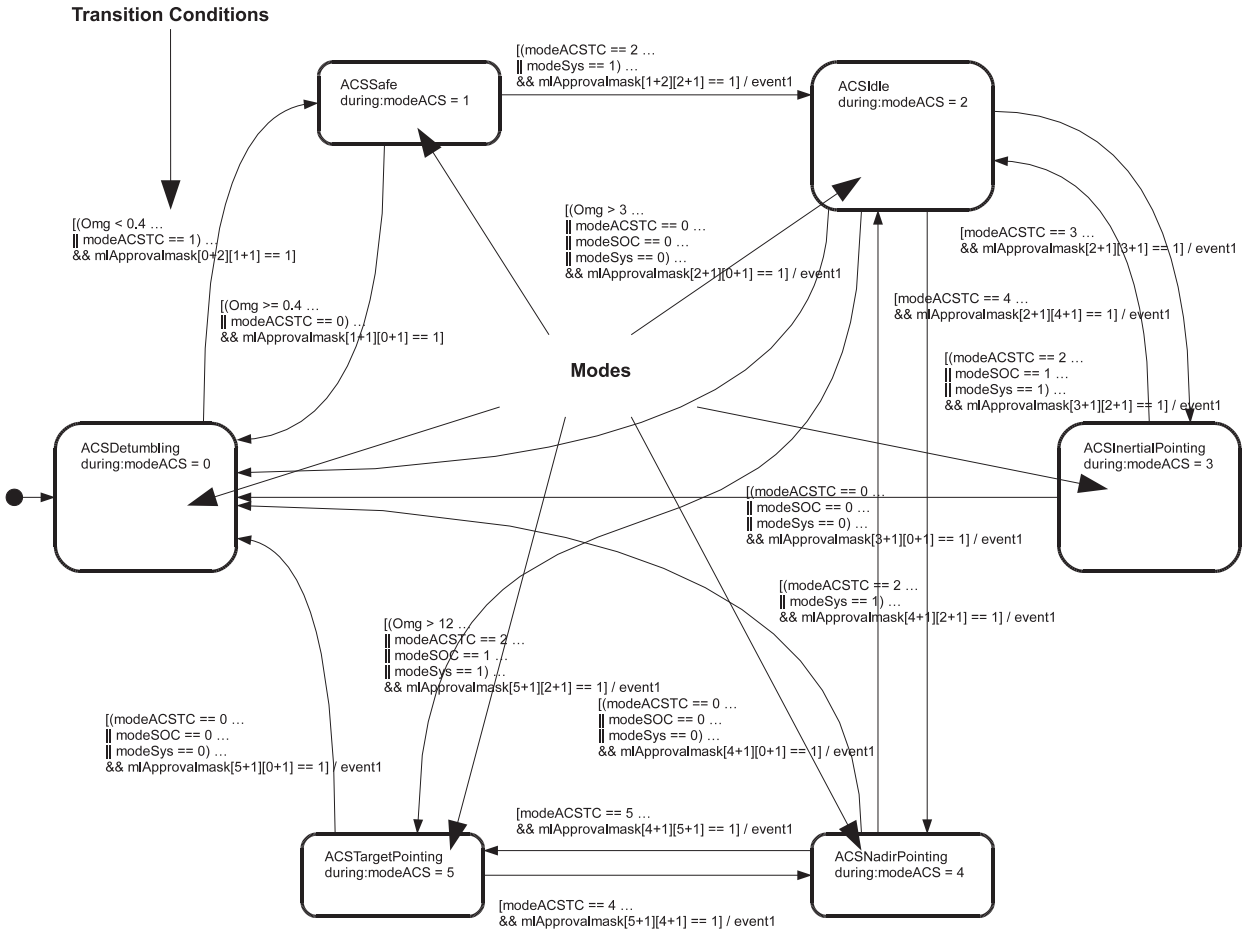

Fig. 6. Attitude control system Stateflow model.

2.4. Implementation in FPGA

The best way to maximize the computational capability of the FPGA is to parallelize satellite on-board processes, such as the attitude control process and house keeping process, which are usually executed in serial periodically. The designed configuration of the parallel control algorithms of the on-board computer is illustrated in Fig. 7 [5]. It can be seen that the entire control algorithm is divided into horizontal and vertical directions running in parallel. The application level control algorithms, which are relevant to this study, are responsible for controlling subsystems. The developed state-machine of each subsystem can be implemented into each logic block in the figure. Because these control algorithms run totally in parallel, those logic blocks representing subsystems shall communicate with each other so that the combination of all modes remains consistent. The state-machine developed and tested in the environment mentioned above can be implemented into FPGA with minimum modification.

2.5. Contingency concept

The contingency operation of the *Flying Laptop* is classified into two cases: contingency levels 1 and 2. In case of level 1 contingency, the satellite system goes into the idle mode. For example, in case of insufficient battery charge for a planned operation, the satellite goes into idle mode. The level 2 contingency represents severe contingency cases in which, for example, a primary sensors for idle mode attitude determination and control has been broken and the idle attitude control becomes impossible. In case of level 2 contingency, the satellite system goes into the safe mode. Every time the system goes into the safe mode, the attitude control system goes through the detumbling mode so that the rotational rate remains under a certain limit (Fig. 3),

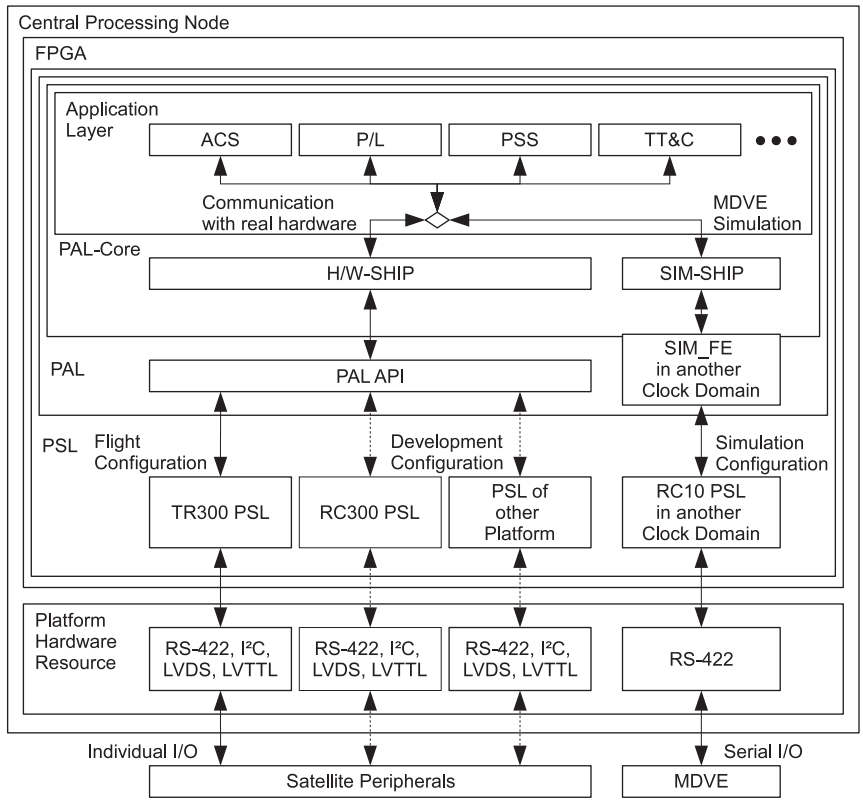

Fig. 7. FPGA control algorithm configuration.

preventing the payload camera systems from pointing the sun directly. During the safe mode, the causes of the contingency are analyzed and identified on a ground basis and the system moves back into the idle mode by telecommands.

2.6. Sample scenario

In this section a sample operational scenario is described. An exemplary case of the BRDF measurement is listed in Table 2. To perform the BRDF measurements, the satellite points the camera systems towards the target on the Earth surface during the fly-by. The camera systems obtain images from different observation angles along the path. Before and after performing BRDF measurements, the system is in idle mode and the attitude control system performs idle attitude control mode. In order to keep this sample scenario brief, an assumption was made that the other subsystems remain in the same mode during the measurement.

As listed in the table, the modes of payload and attitude control system transit from one to another, which results in system level combinational functions letting the system behave as a single satellite performing the desired mission operation. It is important to note that these two subsystems are running in parallel. In this sense, the system modes are products of a combination of subsystem modes and are not the dominating input parameter into the subsystems.

Table 2
Sample operational scenario.

Transition number	System/ subsystem	State/state transitions
0	System	Idle mode
1	System	Go to active operation mode
2	P/L	Go to stand-by mode
3	ACS	Go to nadir pointing mode
4	ACS	Go to target pointing mode
5	P/L	Go to image acquisition mode
6	P/L	Go to idle mode
7	ACS	Go to nadir pointing mode
8	System	Go to idle mode
9	P/L	Go to off mode
10	ACS	Go to idle attitude control mode

2.7. Commanding concept

The commanding concept of the *Flying Laptop* differs based on the operational phase. At the beginning of the mission life time, the satellite is operated only by telecommands without any on-board intelligence. The sample scenario described in the former section is therefore performed by sending commands one by one. Each transition number represents the corresponding command. The commands are time-stamped and executed at the defined on-board time.

At the later operational phase, after having the functional performance of all components tested, the autonomy level of commanding can be raised. In this case, the satellite uses tables of operational sequences, and commands in the above table are executed by a single telecommand. For the sample scenario described in Section 2.6, a single telecommand "BRDF measurement" with appropriate parameters, such as target positions and camera settings, is up-linked.

3. On-board data handling

3.1. Data volume

The data to be handled aboard the satellite is composed of housekeeping and science data. Housekeeping data is continuously collected over time and constitutes a negligible part of the total data volume, while science data amounts to notable volume depending on the payload sensors, the communication concept of the satellite, and its scientific usage. The payload sensors of the *Flying Laptop* determining the data volume are listed in Table 3 [6]. Instruments producing a negligible amount of data or are communication devices themselves are not taken into account in the following.

The design-driving application for the on-board storage capacity is the BRDF experiment. It has the highest priority among the other scientific Earth observations. This measurement requires 200 images per channel taken by the multi-spectral imaging camera system (MICS). With a radiometric resolution of 12 bit and 1.05 MPixels, one measurement amounts to 945 MB of raw data. Based on this number, the minimum storage capacity was defined to 1 GB even though all science data will be compressed on the satellite and compression rates are expected to be fairly high for the BRDF measurement data.

Table 3
Camera system characteristics.

Camera	MICS	TICS	PAMCAM
Spectral bands	530–580 nm 620–670 nm 820–870 nm	8–9 μm 10–12 μm	400–800 nm
Pixel resolution	3×1024×1024	320×240	1280×1024
GSD (m)	25	100	200
Swath width (km)	25	32	250
Digitalization (bit)	12	14	12

For other Earth observation applications this capacity is rather small. The continuous Earth surface scanning along the ground track would require much more capacity than available. For this application MICS generates data at 1.6 MB/s, TICS at 0.05 MB/s and PAMCAM at 0.06 MB/s, which add up to about 1.7 MB/s. Assuming a continuous scanning in nadir pointing except for the time when the satellite travels in the Earth's shadow, the amount of uncompressed raw data adds up to about 6.5 GB per orbit. The satellite is operated by a single ground station located at the IRS so a maximum gap of eight orbits between two subsequent ground station accesses can occur. Assuming a compression rate of about 2.0 for multispectral images of the Earth surface [7], an on-board storage capacity of 26 GB would be needed.

The storage of such a data volume is expensive in terms of mass and space requirement, power consumption and price. Moreover, it is hard to down-link and even to store and handle on ground for a small satellite project. Consequently, intensive data processing is required as described in Section 4. The actual storage size of the *Flying Laptop* is planned to be 4 GB.

3.2. Data storing

The on-board computing system of the *Flying Laptop* is a single centralized OBC which consists of four central processing nodes (CPNs) in a redundant configuration and one command decoder and voter (CDV) [8]. The CPNs are based on reconfigurable SRAM FPGAs and the CDV selects one master node out of the four. The mass memory is mounted on the CDV and can be accessible from each CPN. The above mentioned control algorithm is implemented into CPNs and the CDV only has preliminary functions such as voting and memory management. The camera systems are controlled by the master node CPN and the obtained image data is processed within the node. Either pre-processed or raw data are transported to the CDV through the point-to-point communication lines between them and stored into the mass memory on the CDV. During ground contacts, the stored data are transported from the CDV to the master node, coded and down-linked to the ground station. The stored data in the mass memory can be read

Table 4
Communication channels.

Channel	Baud rate (bps)	Operational concept
VHF up	19.2 K	On demand
UHF up	4.8 K	Periodical/on demand
UHF down	38 K	Periodical/on demand
S up	19.2 K	Primary
S down low gain	150 K	Primary (non-directional)
S down high gain	2.2 M	Primary (directional)
Ka up	10 M	Max. 2 times/day
Ka down	< 500 M	Max. 2 times/day

from the nodes and processed during the non-ground-contact phase.

3.3. Configuration management of FPGA hardware logic

The configuration files for the hardware logic of the SRAM FPGAs are stored in the triple-redundant flash-memories on each CPN. Every time a CPN is powered on or reset, the configuration file is read into the configuration memory element inside the FPGA. The contents of the flash-memories can be replaced with the up-linked new configuration files directly by the CDV through a dedicated communication line. In case of failures of these flash-memories or if required the hardware logic of each SRAM FPGA can also be reconfigured by the CDV in a streaming mode. The details of this are beyond the scope of this paper, and will be the subject of a separate publication.

3.4. Communication concept

The communication channels of the *Flying Laptop* are listed in Table 4 with their baud rate and operational concept. The primary communication channel of the satellite is S-band. The LG and HG in the table stand for low gain and high gain, respectively. The low gain down-link channel uses omni-directional antennas and can be used in case the satellite does not perform pointing attitude control towards the ground station. A patch antenna for the high gain down-link channel is mounted on the same surface as the camera systems and can be used only when the satellite performs pointing attitude control. Unless the system is in safe mode, the satellite is able to perform high gain communication. The UHF and VHF channels are intended for the cooperation with other universities and amateur band communities and used as back-up channels of the S-band. The Ka-band up and down-link is an experimental channel and can transmit up to 500 Mbps in down-link. Even taking the error correction coding into account, this channel provides a sufficient bandwidth for large amounts of stored data during a single path. Modestly assumed, an effective transmission rate of 300 Mbps in 12 min can achieve 27 GB. Theoretically, the data obtained by the continuous Earth surface scanning can be down-linked during one ground station access.

4. On-board payload data processing

4.1. Purpose and benefits

There are two purposes for on-board payload data processing. Firstly a reduction of data volume to be stored and transmitted, and secondly the need for on-board processed data for certain applications. The necessity for data reduction results from the enormous amount of data generated by latest sensors due to high spectral, spatial, temporal, and radiometric resolution. Either the on-board storage capacity or the amount of down-link rate or down-link time represents usually the bottle neck. Besides restricting scientific observation time there are mainly three solutions to overcome this issue. One can remove redundancies from the data, analyze the data or screen the data.

Removing redundancies can be performed by various types of compression. As this is the only way to preserve all the raw sensor data, its implementation is self-evident especially if the data volume is large, storage capacity is tight and computational capabilities are at hand. Analyzing the data directly on the satellite not only requires high computational performance and long implementation time, but also takes the risk of losing important information because the raw data cannot be restored, while it decreases the needed storage capacity decisively. Moreover, the application of the analysis has to be very specific so a wide usage of the data is not feasible. The last possibility to reduce data volume is to screen the incoming data and select only valuable information for further processing, storage and down-link. A relatively simple image screener could be a cloud detection algorithm which marks all incoming images as non-valuable when they exceed certain cloud coverage. The *Flying Laptop* shall be able to perform continuous nadir observations where all incoming images are pre-processed, evaluated in regard to their relevance, and afterwards either stored or discarded.

As any processing on the FPGA will work in real-time this offers additional applications. For example, a real-time observation, processing, and down-link in a streaming mode is planned for the *Flying Laptop*.

This application can be used for real-time control of the satellite, as well as for disaster monitoring. The satellite observes a defined area, processes the images and transmits them as a standard image format to a ground station in or near that area. In this way cooperation partners and even amateur radio communities can access live images from their surrounding landscape without the need for any processing.

But the benefits of on-board processing do not come without disadvantages. All computations require the corresponding infrastructure on the satellite. For the FPGA-based OBC on the *Flying Laptop* this means that there are fixed amount of hardware logic available. Thus, the number and complexity of algorithms, which can be mapped on the hardware, is restricted.

4.2. Image processing algorithms

In order to perform meaningful on-board processing the following algorithms will be implemented on the *Flying Laptop*. Radiometric and geometric sensor correction algorithms account for the sensors' non-idealities, followed by coregistration of the three MICS channels and the two TICS channels. Image screening algorithms can now be applied to distinguish valuable images from non-valuable ones. An FPGA implementation of a cloud detection algorithm has been demonstrated by Williams [9]. Depending on the application this can also be done for water, ice, sand, or similar homogeneous surfaces. Once the image has been accepted, it will be georeferenced. Rectification can generally be neglected in nadir pointing mode as the field of view spans only 2°. For large nadir offset angles, a basic rectification accounting for the Earth's curvature and geometric distortion of the camera can be applied. As no digital elevation model is available on-board this is only reasonable for flat terrain.

The next steps are application dependent. For land cover classification a simple supervised classification algorithm such as the maximum-likelihood algorithm will initially be implemented. For a better performance a neural network classifier [10] or neuro-fuzzy classifier [11] will be of interest. After compression the raw data is stored in the flash memory for down-link. JPEG-LS is foreseen for any lossless compression of the image data.

4.3. Concept implementation

To benefit from the FPGA technology one must utilize its parallel processing capabilities. This means that the algorithms mentioned above are running in parallel,

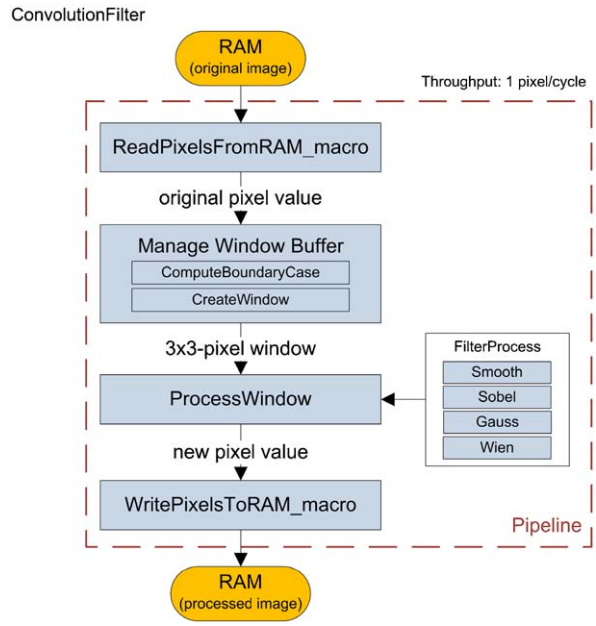

Fig. 8. Pipeline process example—convolution filter.

but also requires an optimized implementation of the algorithms themselves. They are programmed as pipeline processes, which is a common design technique for FPGAs to achieve high throughput and parallelism [12]. Fig. 8 shows an example of the implementation of a convolution filter algorithm, which can be used for edge detection, denoising, smoothing, sharpening and similar applications. It consists of four stages which run in parallel. As they run in the same clock domain this is called synchronous pipelining. Each stage hands its output directly over to the next stage and receives new input at the same time. This concept enables a high data throughput but makes high demands on the programming as the clock rate is limited by the longest combinational delay in the pipeline. In the present example a throughput of one pixel per clock rate is achieved once the pipeline is primed. To prime the pipeline for an image with n columns and m lines roughly $2n$ cycles are necessary. For the processing of the whole image, $(m+2)n$ cycles are needed. At a clock rate of 200 MHz the processing of around 200 images with 1024×1024 pixels could be performed in a second.

Any algorithm which has not yet finished processing of the current image when the next one arrives will limit the real-time capabilities of the system. Thus, such complex algorithms can be mapped more than once on the FPGA and run in parallel. While one image is processed in the first instance of an algorithm, another instance will process a second image. Each type of algorithm

has a common queue in the form of a FIFO which is accessed by multiple instances of this algorithm. Once one instance has finished processing its current image, it picks the next task from the queue. In this way the number of the instances can be easily adjusted depending on the need for real-time performance. A trade-off study has to be performed once the required number of cycles for each algorithm type is known. The FPGA features only a restricted number of gates so the number of algorithm instances is limited.

So far, only basic image processing operations such as radiometric calibration, image warping and various convolution filters have been implemented and tested on an FPGA test board.

5. Trade-off on on-board autonomy

There are several grades of on-board autonomy foreseen for the *Flying Laptop* as follows:

- commanding level,
- intelligent image processing, and
- self decision making.

The first autonomy represents the application of high level commands, as also described above. At the beginning of the mission phase, the satellite shall be operated by low level commands only. After the components have been tested in orbit, the autonomy level can be raised by sending high level commands. Those high level commands are translated to sequential low level commands on-board. The second autonomy is the intelligent processing of the payload data. By means of the implemented data processing algorithms, the satellite selects which data shall be stored and how it shall be processed. This type of on-board image processing plays an important role to achieve any high level on-board autonomy. The third autonomy is the highest and represents the on-board intelligence, enabling the satellite to decide by itself which mission to perform. In this case, the satellite should autonomously decide which observations are of interest. A simple example could be multiple sequential BRDF measurements. Depending on the current cloud coverage above the target, the satellite decides whether or not it performs target pointing, whether or not it stores the data and whether or not it proceeds to observe the next target depending on the remaining storage capacity. For that purpose, extended image screening algorithms have to be developed based on the second autonomy. Another example will be the case that during the continuous Earth surface scanning, the satellite can also decide to perform a target pointing if a target of interest is discovered. This would require complex object detection algorithms which might be based on neural networks.

The benefits of on-board autonomous processing are clearly the reduction of data volume and the employment of the satellite for real-time applications. Moreover, cooperations with other institutions are simplified as entire data products can be delivered by the satellite without taking the detour over the IRS ground station. In contrast, processing on ground has an advantage that all raw data is still available after the processing, so changing or correcting the analysis algorithms remains always possible. Another benefit of on-ground processing is certainly the circumstance that analysis routines are often readily available. At least, they are easier to implement as there are virtually no restrictions on ground in terms of computational power, storage capacity and the type of hardware and software implementation used. Hence, the implementation of complex analysis routines on ground is faster, more cost effective and even more reliable. But still, if real-time data is needed or the down-linkable data volume is the bottle neck, on-board processing becomes indispensable.

6. Conclusions

In this paper, the operational design and the on-board payload data processing of the small satellite *Flying Laptop* are described. The designed operational plan maximizes the advantage of the parallel processing capability of the FPGA-based on-board computer, enabling the achievement of a variety of technology demonstration and scientific Earth observation missions. The designed architecture of the system mode is arranged in a hierarchical manner decomposed into subsystem and components modes. This is modeled, developed and tested in the MathWorks Simulink-/Stateflow Toolbox environment. The developed state-machines are then implemented into FPGA hardware logic. A sample operational scenario and the corresponding commanding strategy are described.

Also described are the developed payload data processing methods and procedures together with the related requirements such as data volume, storage capability, and down-link capabilities. The data processing algorithms, implemented as pipeline processes, enable on-board real-time processing, which expands the scientific usage of the data and offers new possibilities for on-board autonomy. After having these algorithms developed and tested, the operational concept

will be adapted according to their actual performance capabilities.

Finally, the on-board autonomy foreseen for the *Flying Laptop* is described and the trade-off between on-board autonomy and ground control is discussed.

Through this study, it became clear that additional ground stations, including amateur ground stations and those at other universities bring tremendous advantages in terms of efficient Earth observation and data acquisition by the *Flying Laptop*. The *Flying Laptop* Team is also still looking for new applications for the *Flying Laptop* where its strength in computational performance can be demonstrated. To achieve the greatest benefit of the satellite's capability, cooperations and new applications for the satellite are always very welcome.

References

[1] R. Laufer, H.-P. Röser, et al., Lunar Mission BW1—a university small satellite mission to the Moon, in: Proceedings of Sixth International Conference on Exploration and Utilization of the Moon, India, Udaipur, November 22–26, 2004.

[2] T. Kuwahara, F. Huber, A. Falke, M. Lengowski, S. Walz, G. Grillmayer, H.-P. Röser, System design of the small satellite Flying Laptop, as the technology demonstrator of the FPGA-based on-board computing system, in: Proceedings of 58th International Astronautical Congress, Hyderabad, India, September 24–28, 2007.

[3] M. von Schönermark, B. Geiger, H.-P. Röser, Reflection Properties of Vegetation and Soil with a BRDF Data Base, first ed., Wissenschaft und Technik Verlag, Berlin, 2004, ISBN 3-89685-565-4.

[4] G. Grillmayer, Use of new developments of attitude control sensors for the micro-satellite Flying Laptop, in: Proceedings of 57th International Astronautical Congress, Spain, Valencia, October 2–6, 2006.

[5] T. Kuwahara, A. Falke, C. Ziemke, Y. Muhammad, J. Eickhoff, H.-P. Röser, Development of a hardware-in-the-loop simulation environment on a MDVE for FPGA-based on-board computing systems, in: Proceedings of 26th International Symposium on Space Technology and Science, Hamamatsu, Japan, June 2–6, 2008.

[6] A. Falke, S. Walz, G. Grillmayer, H.-P. Röser, LED in-flight calibration and model-based development of ACS algorithms for the university micro-satellite Flying Laptop, in: Proceedings of Fourth Small Satellite Systems and Services Symposium, Chia Laguna, Italy, September 25–29, 2006.

[7] T. Vladimirova, et al., On-board compression of multispectral images for small satellites, in: Proceedings of Geoscience and Remote Sensing Symposium, Denver, Colorado, USA, July 31–August 4, 2006.

[8] F. Huber, P. Behr, H.-P. Röser, S. Pletner, FPGA based on-board computer system for the Flying Laptop micro-satellite, in: Proceedings of Data System in Aerospace Conference, Naples, Italy, May 2007.

[9] J.A. Williams, et al., FPGA-based cloud detection for real-time onboard remote sensing, in: Proceedings of IEEE International Conference on Field-Programmable Technology, Hong Kong, December 16–18, 2002.

[10] N.D. Ritte, et al., Application of an artificial neural network to landcover classification of thematic mapper imagery, Computers and Geosciences, 1990.

[11] S.G. Lee, et al., A neuro-fuzzy classifier for land cover classification, in: Proceedings of IEEE International Fuzzy Systems Conference, Seoul, Korea, August 22–25, 1999.

[12] J.L. Hennessy, D.A. Patterson, Computer Architecture: A Quantitative Approach, Elsevier Ltd., Oxford, 2006.